Fast Burrows Wheeler Compression Using All-Cores

Aditya Deshpande* and P J Narayanan

Centre for Visual Information Technology
International Institute of Information Technology
Hyderabad

ASHES, 2015

*Now at the University of Illinois at Urbana Champaign

IIIT Hyderabad

Outline

- **What?**
  - Use all-cores (CPU + GPU) for a common end-to-end application
  - Our focus: **Burrows Wheeler Compression (Bzip2)**

- **How?**
  - Use fast GPU String Sort [Deshpande and Narayanan, HiPC'13]
  - Domain specific techniques for GPU BW Compression
  - All-core framework to use both CPU and GPU together

- **Why?**
  - Commodity computers have multi-core CPU + many-core GPU
  - All-core end-to-end applications help end user leverage them both

Previous Work

- **Multi-core CPUs** (Coarse/Task Parallelism) –
  - LU, QR, Cholesky Decomposition, Random PDF Generators etc.
  - FFT, PBzip2, String Processing, Bioinformatics, Data Struct. etc.
  - Intel MKL and other libraries

- **Many-core GPUs** (Fine/Data Parallelism) –
  - Scan, Sort, Hashing, SpMV, Lists, Linear Algebra etc.
  - Graph Algorithms: BFS, SSSP, APSP, SCC, MST etc.
  - cuBLAS, cuFFT, NvPP, Magma, cuSparse, CUDPP, Thrust etc.

- The focus is typically not **end-to-end** and/or **all-core** applications

Burrows Wheeler Compression

End-users compress/de-compress files on daily basis. Best compressor BW Compression (or Bzip2), a three step procedure:

1. **Burrows Wheeler Transform**
   Suffix sort and use the last last column ***(Most compute intensive)***

2. **Move-to-Front Transform**
   Similar to run-length encoding **(~10% of runtime)**

3. **Huffman Encoding**
   Standard frequency of chars based encoding **(~10% of runtime)**

I meant what I said
and I said what I meant

From there to here
from here to there
I said what I meant

INPUT

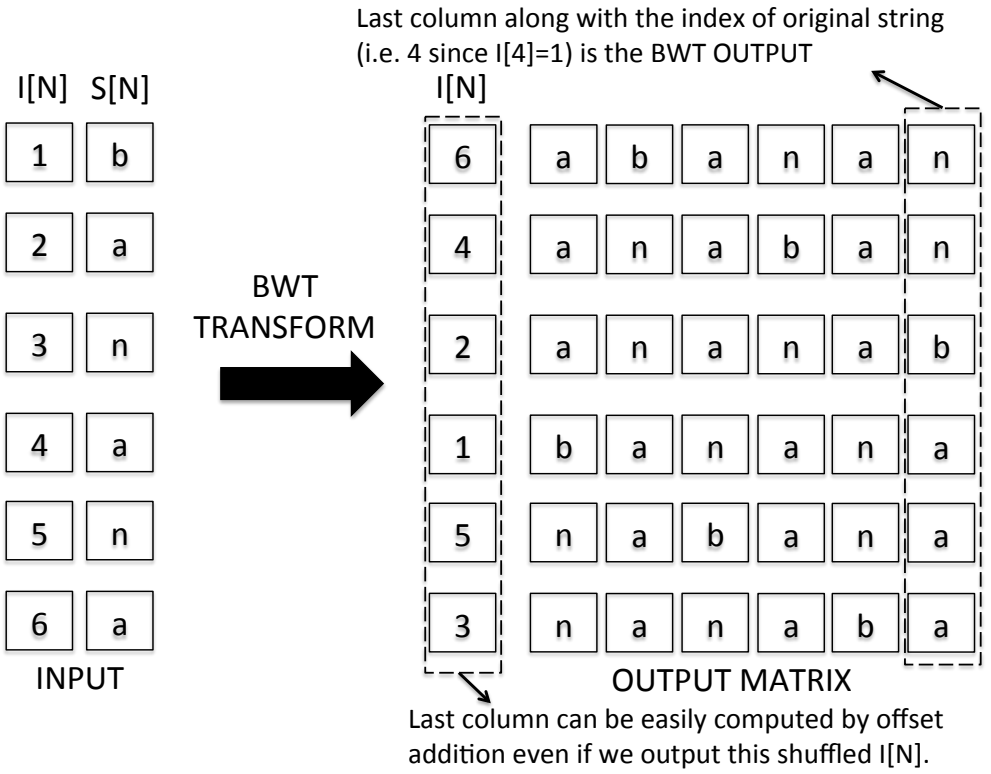
tt
edetttdomIIIIIIomeeddt
sss
eeehhhiniirrrrmmmhhhh
wwwt t aaaoo aaaattrreeeefF nnaaan

After BWT

Amenable to RLE

Burrows Wheeler Transform

- Input String: **I, S**
- Sort all cyclic shifts of **S**
- Last column of sorted strings, with index of original string is BWT
- **O(N)** strings are sorted, each with length **O(N)**
- Suffix sort in BWT has long ties **10^3** to **10^5** characters
- Need a good GPU String Sort that works on longer ties



Sorting

- Textbooks teach us many popular sorting methods

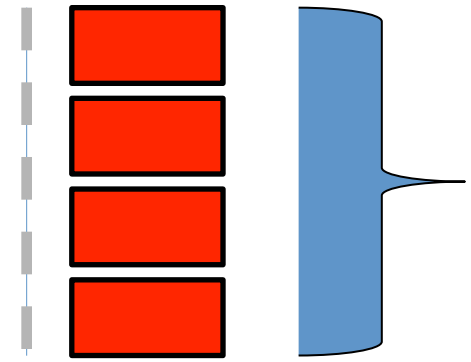
Data is always numbers!

- Real data is beyond just numbers
  - *Dictionary words* or *sentences*
  - *DNA sequences, multi-dimensional db records*
  - *File Paths*

Can we sort strings efficiently?

Irregularity in String Sorting

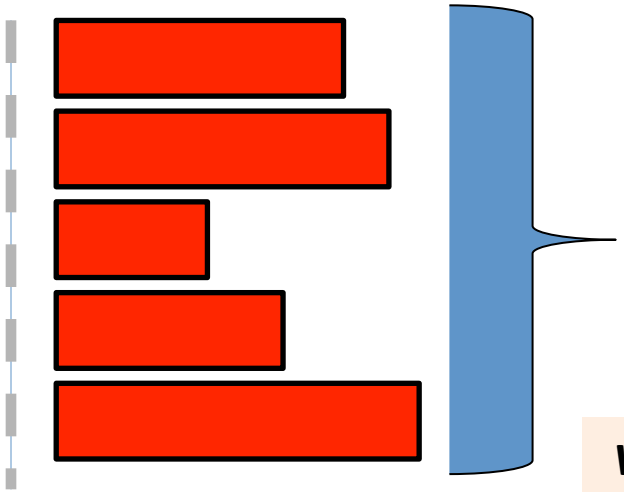
- **Number Sorting (or Fixed Length Sorting)**



FIXED LENGTH KEYS

- Fixed Length Keys (8 to 128 bits)
- Standard containers: *float, int, double etc*
- Keys Fit into registers
- Comparisons take O(1) time

- **String Sorting (or Variable/Long Length Sorting)**



VARIABLE LENGTH KEYS

- Keys have no restriction on length
- Iteratively load keys from main memory
- Comparisons ***not*** O(1) time
- Suffix Sort *(1M strings of 1M length!)*

Variable work per thread and arbitrary memory accesses: **IRREGULARITY**

Previous String Sort

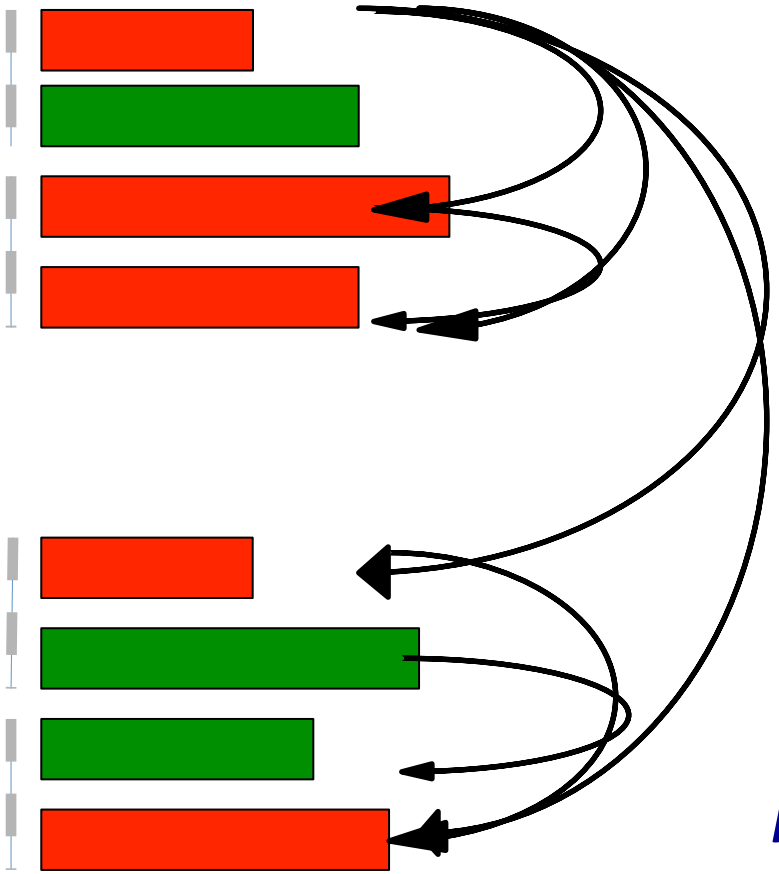
CPU

- Multi-key Quicksort
  [Bentley and Sedgewick, SODA'97]
- Burstsort
  [Sinha et al., JEA'07]
- MSD Radix Sort
  [Kärkkäinen and Rantala, SPIRE'08]

GPU

- Thrust Merge Sort
  [Satish et al., IPDPS'09]
- Fixed/Var. Merge Sort
  [Davidson et al., InPar'12]
- Hybrid Merge Sort
  [Banerjee et al., AsHES'13]
- Our String Sort [HiPC'13]
  (Radix Sort)

Merge Sort: Iterative Comparisons

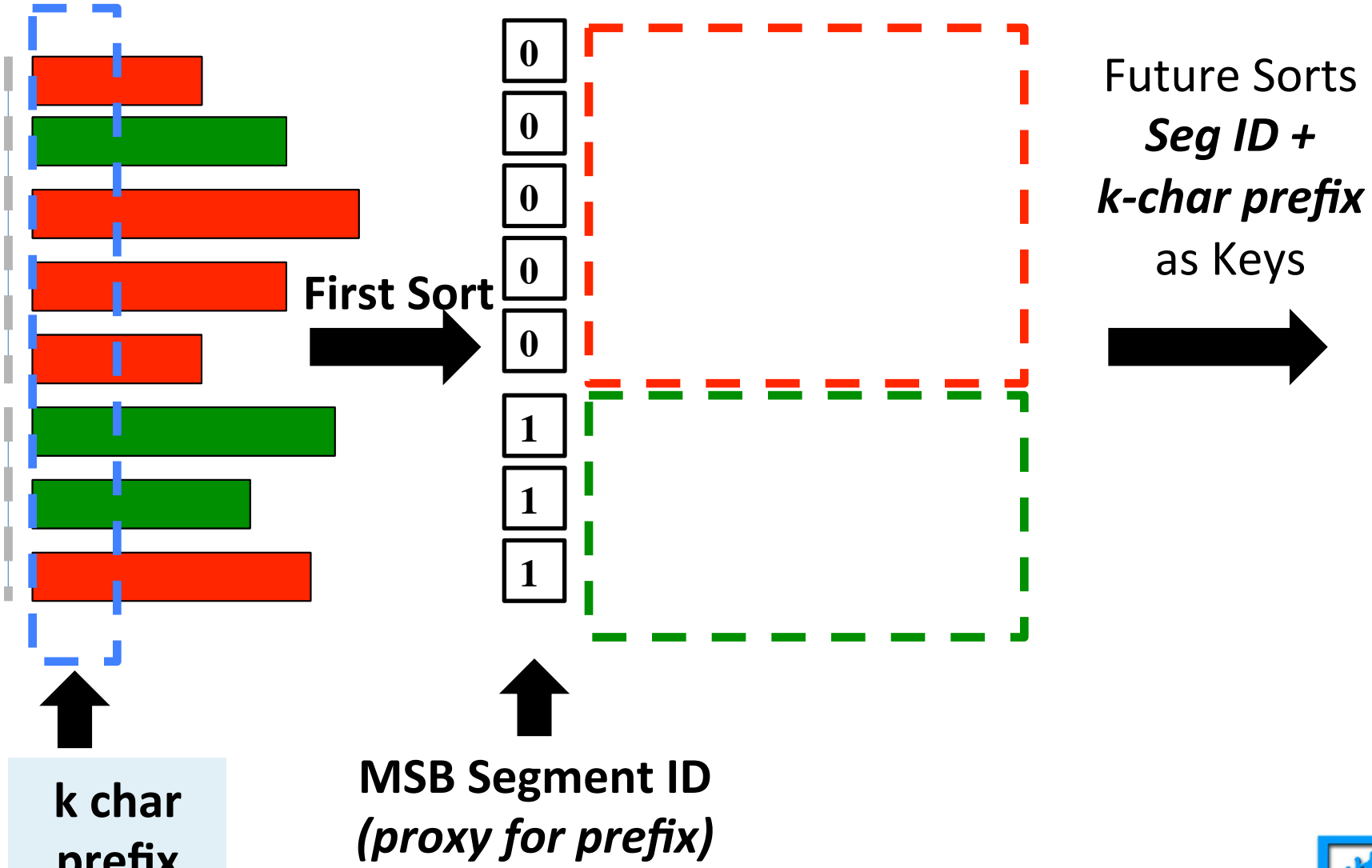


- Repetitive loading for resolving ties in **every** merge step
- Davidson et al. show that *"After every merge step comparisons are between more similar strings"*
- Previous GPU String Sorts are based on Merge Sort

Iterative comparisons = High Latency Global Memory Access = Divergence ☹

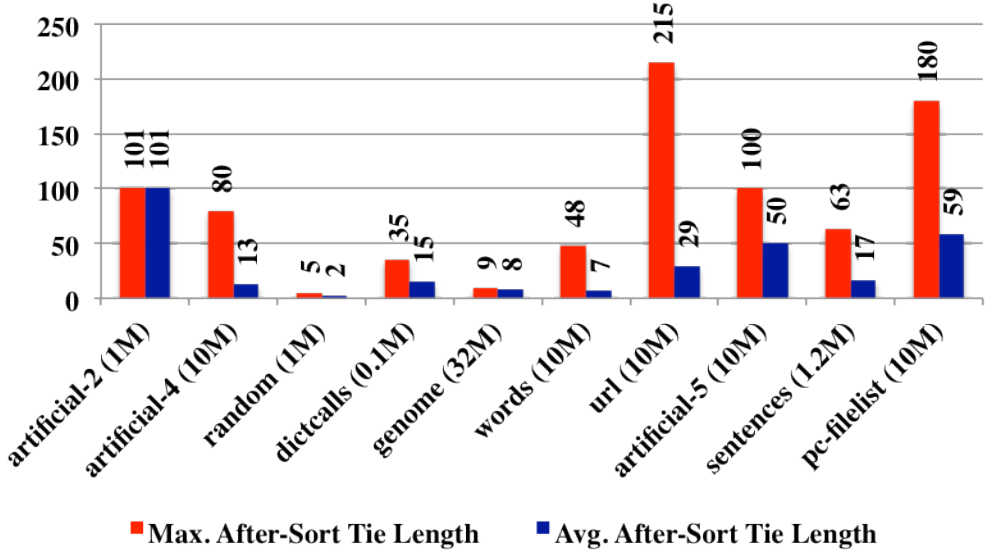
We develop a Radix Sort based String Sort

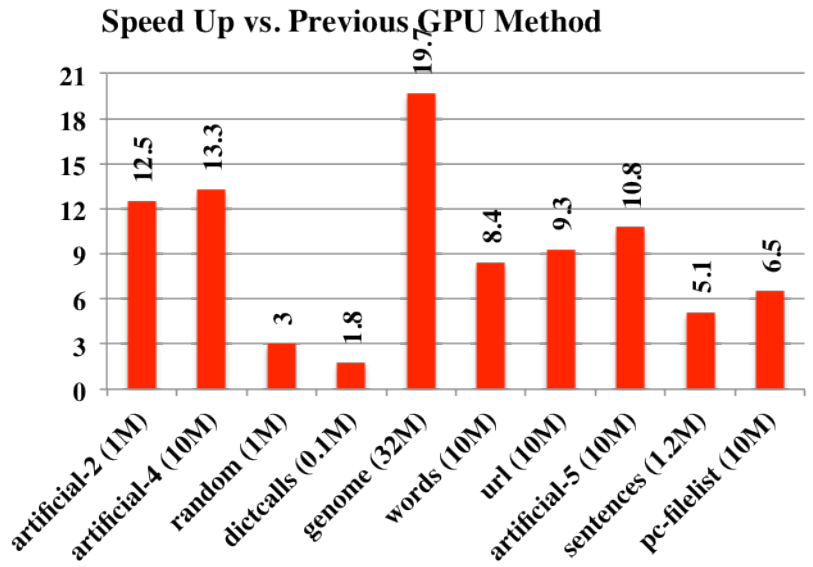
Radix Sort for String Sorting



Results

- **After-Sort Tie Length:** Indicates difficulty of sorting a dataset



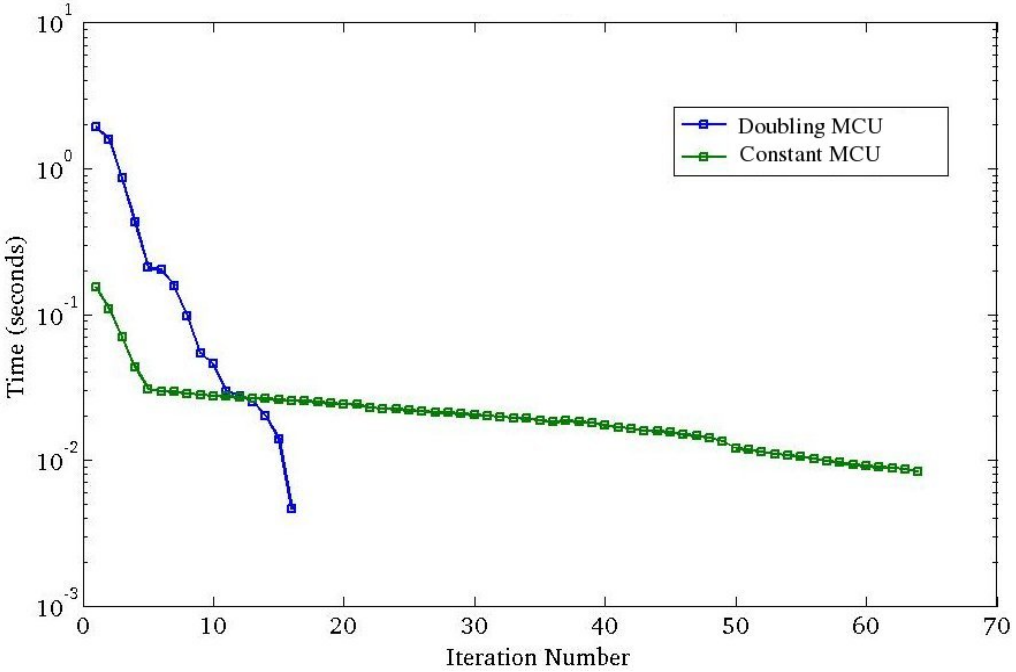


- Suffix sort of BWT has still higher ties and requires many sort steps

- We develop domain-specific sort techniques for BWT

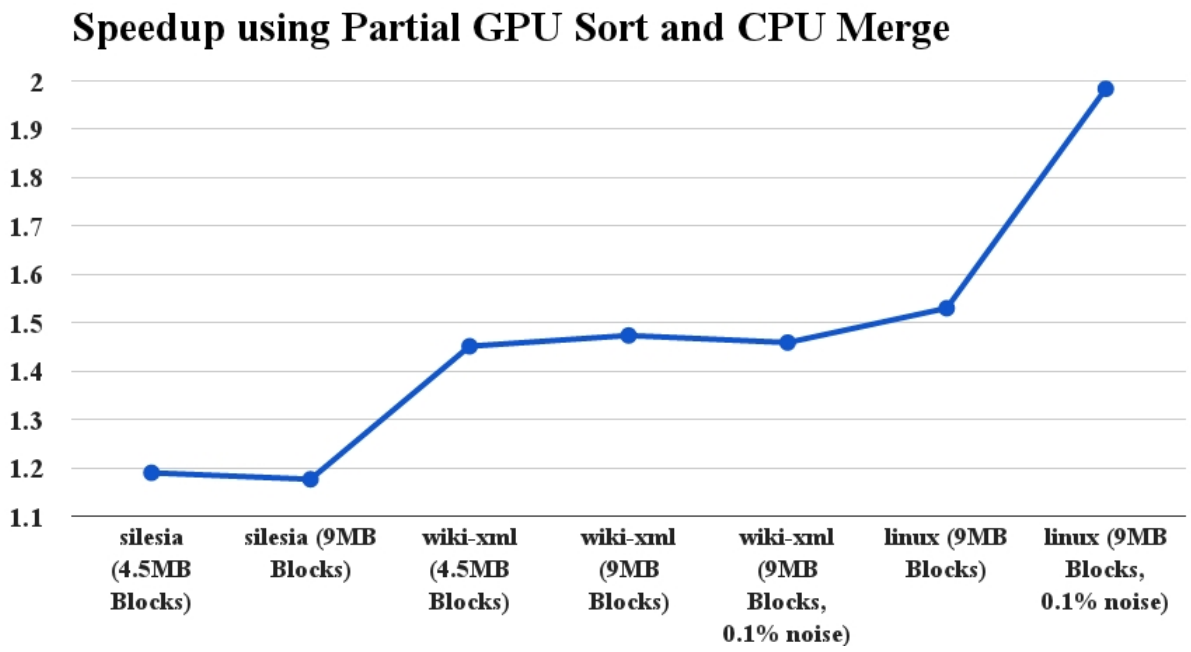
Code from **cvit.iiit.ac.in**

Modified String Sort for BWT



- **Doubling MCU length of String Sort**
  - MCU length determines #sort steps
  - Large #sort steps for long ties and thus, longer runtime ☹
  - Use fixed length MCU initially, then double to reduce sort steps
  - 1.5 to 2.5x speedup

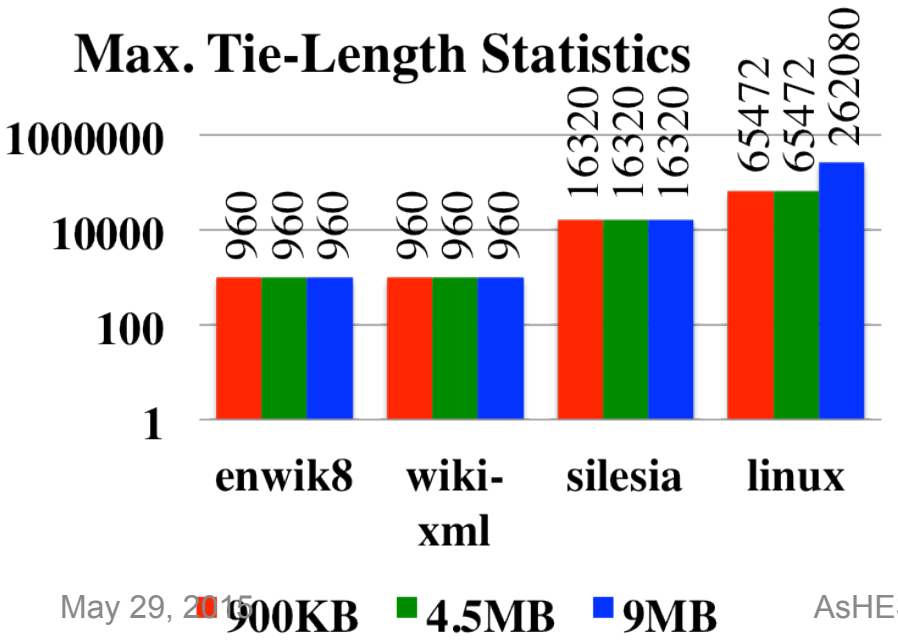
Modified String Sort for BWT

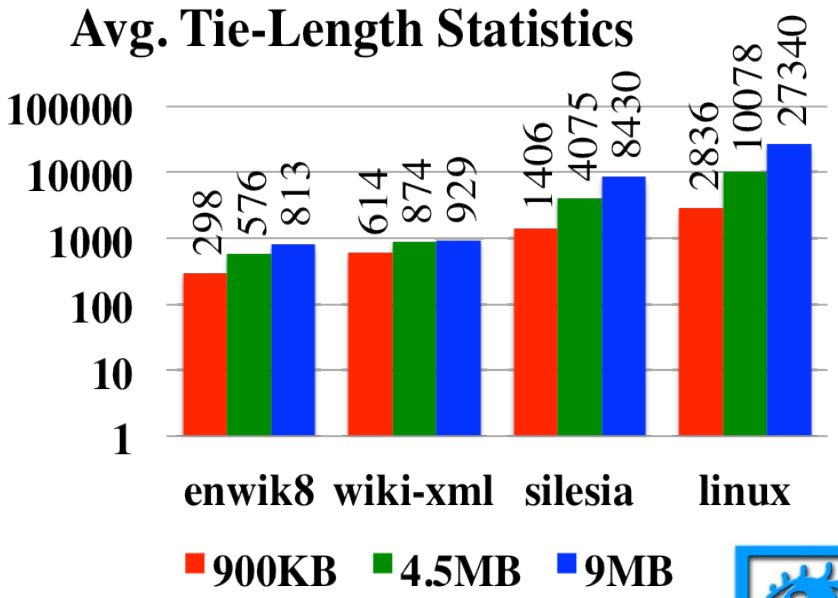


- **Partial GPU Sort and CPU Merge**
  - Cyclically shifted strings have special property
  - We can sort only 2/3rd strings, synthesize rest w/o iterative sort
  - Sort all (mod 3) ≠ 0 strings iteratively
  - 1st char of (mod 3) = 0 string, rank of next in 2/3rd sort enough to sort remaining 1/3rd strings
  - Non-iterative overlapped merge also possible **(CPU)**

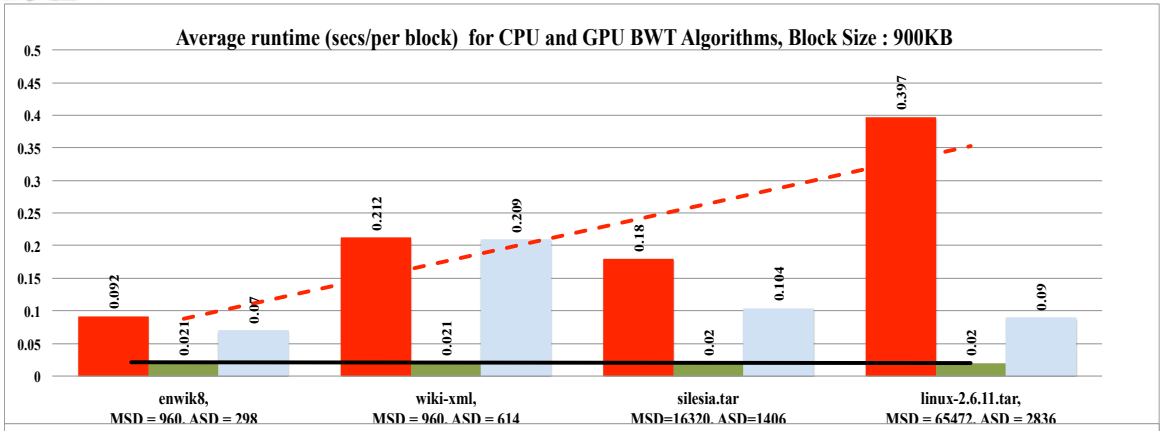
Datasets GPU BWT

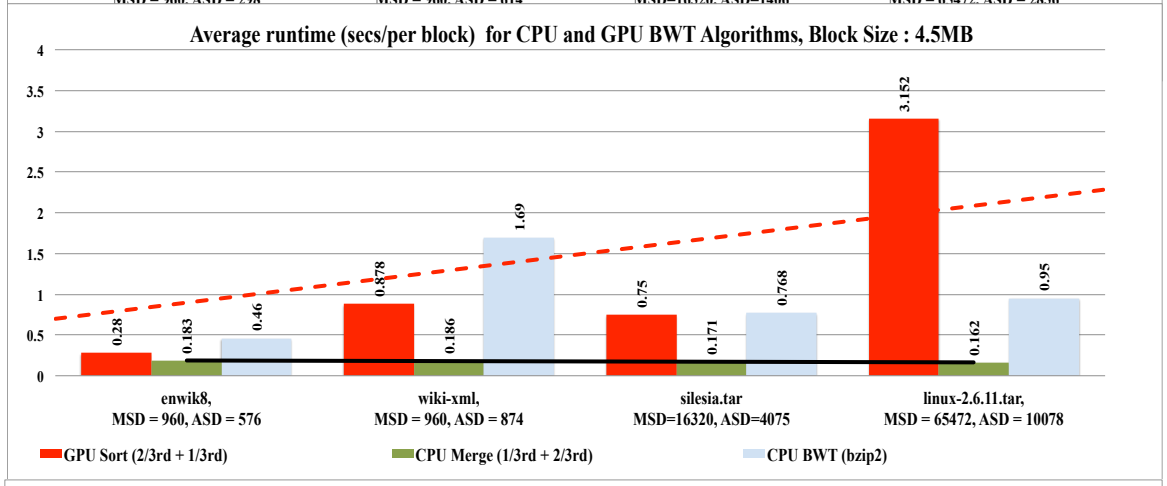
- **Datasets**
  - **Enwik8:** First 10^8 bytes of English Wikipedia Dump (96MB)
  - **Wiki-xml:** Wikipedia xml dump (151MB)
  - **Linux-2.6.11.tar:** Publicly available linux kernel (199 MB)
  - **Silesia Corpus:** Data-compression benchmark (208MB)
- **Tie-Length vs. Block Size**

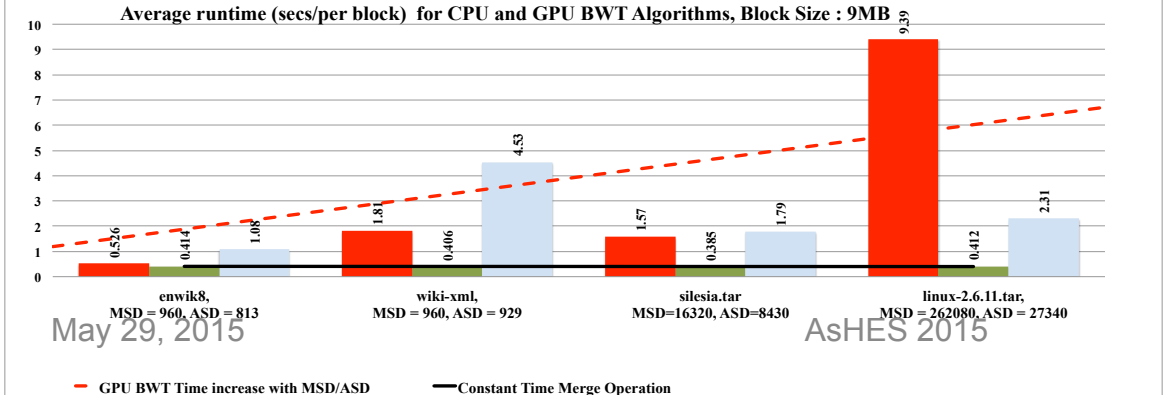




GPU BWT vs. Bzip2 BWT







No speedup for small blocks

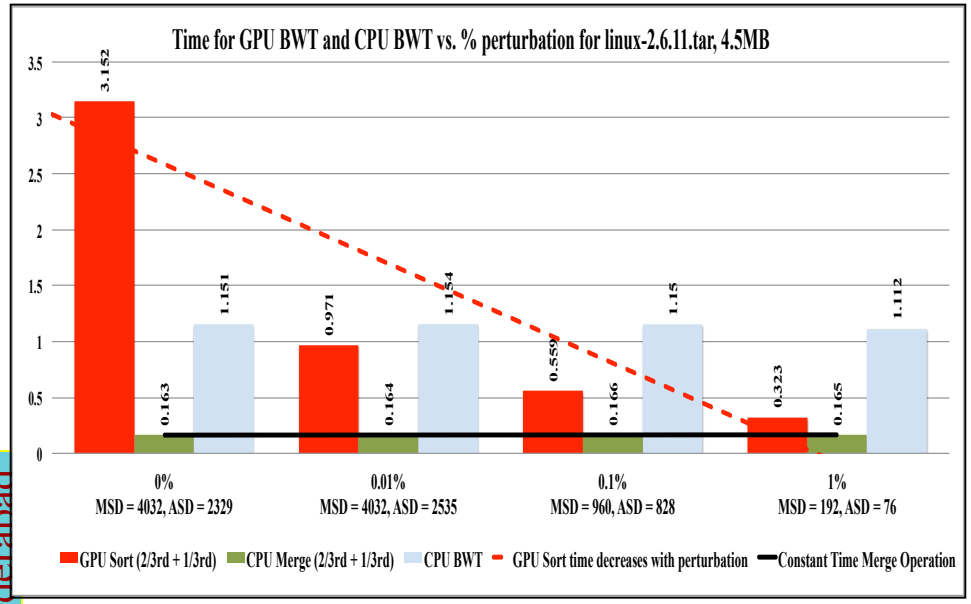
GPU not utilized sufficiently

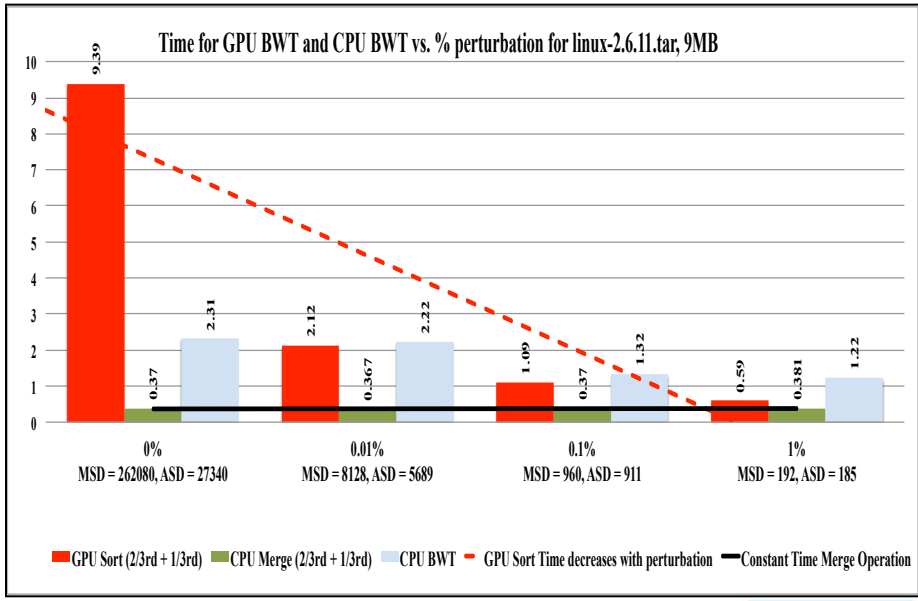
Speedup on large blocks

GPU still slow for worst-case linux dataset

String Perturbation

- Large #sort steps result from repeated substrings/long ties
- Runtime reduces greatly if we break ties
- Perturbation **'add random chars at fixed interval'** to break ties
- Useful for applications where BWT transformed string is irrelevant, and BWT+IBWT are used in pairs (viz. BW Compression)
- Fixed Perturbation can be removed after IBWT

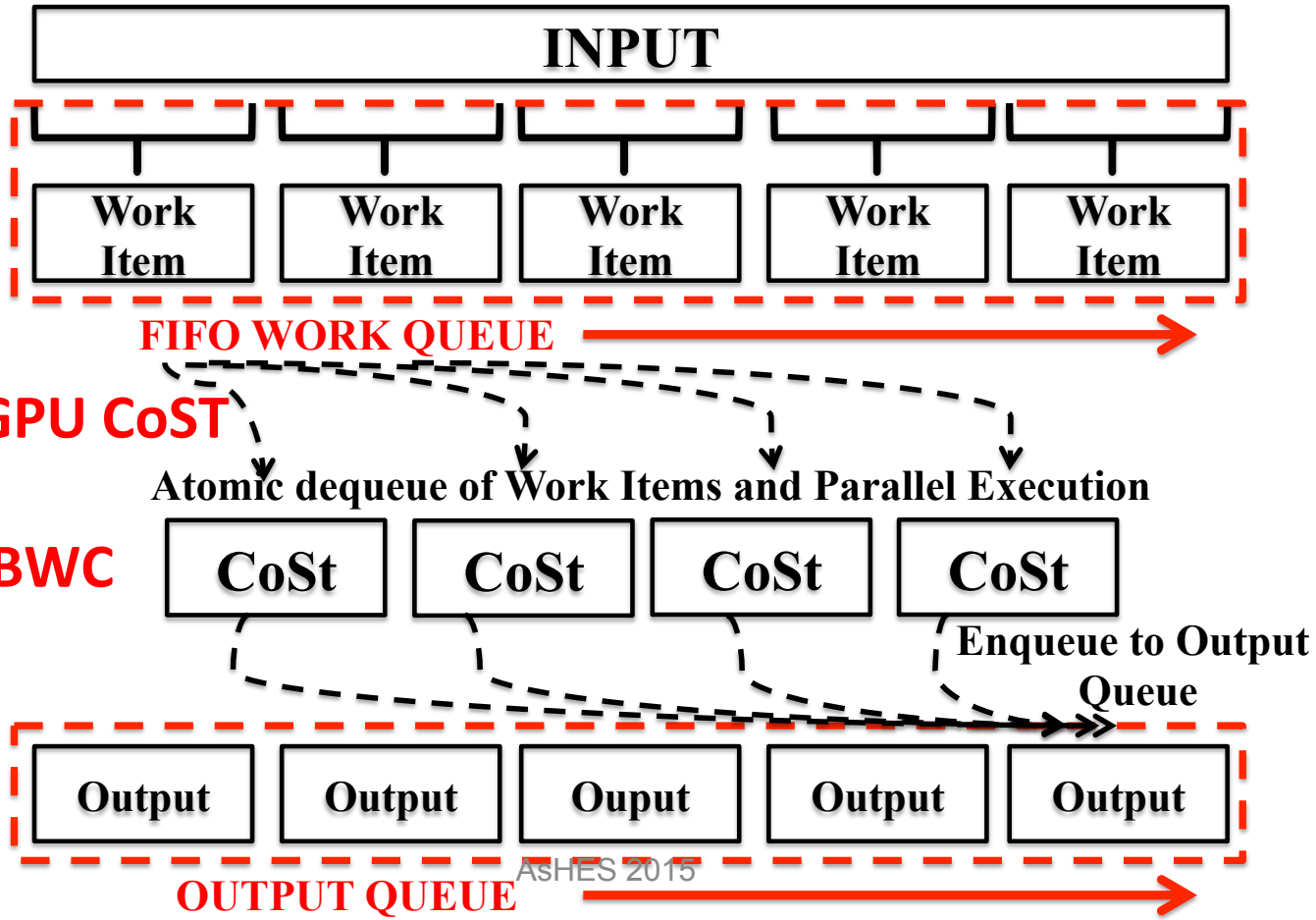




Linux-9MB Blocks, 8.2x speedup with 0.1% perturbation

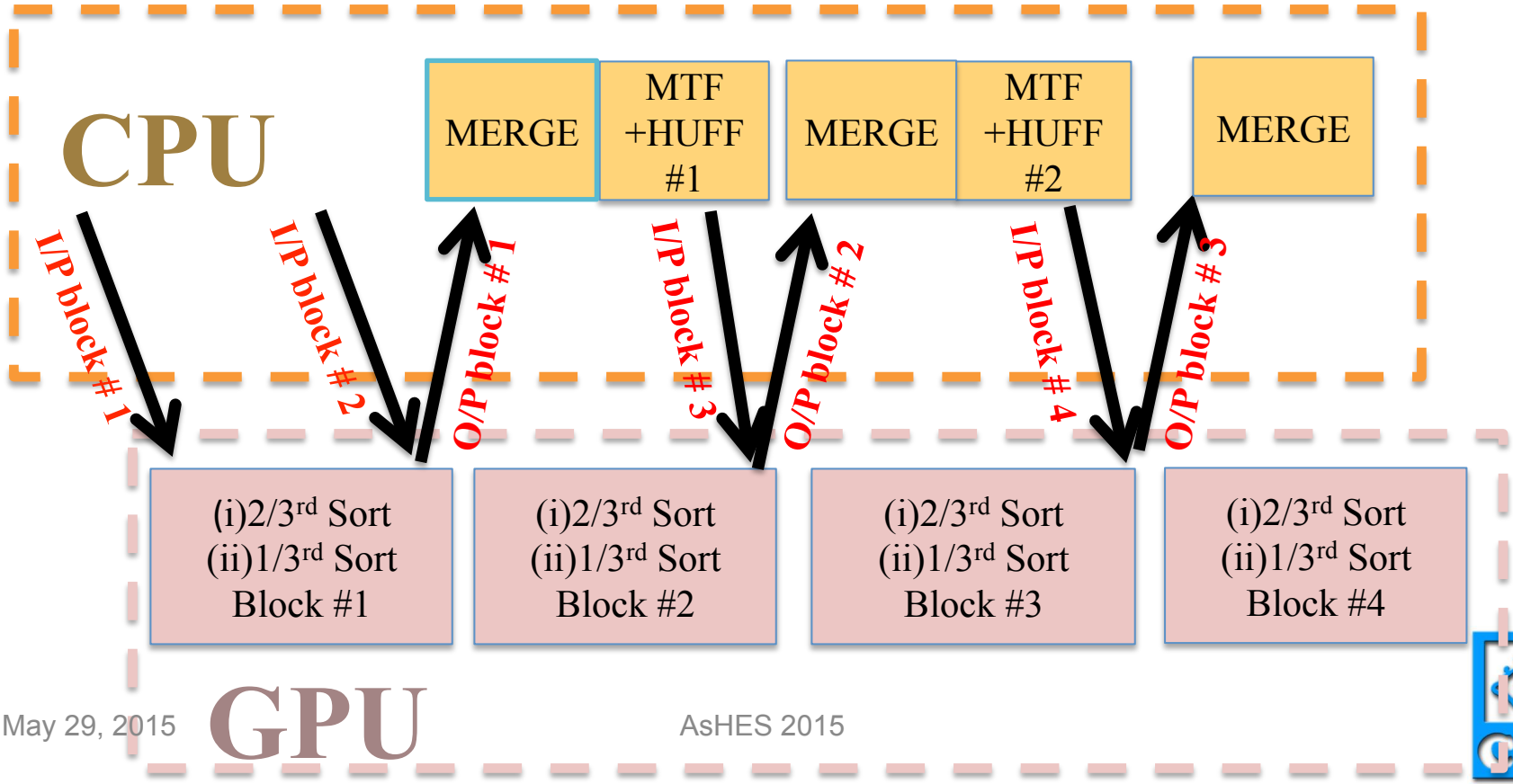
All-Core Framework

- System made of CoSt's:
  - GPU with controlling CPU thread a CoSt
  - Other CPU cores are CoSt's
- Split blocks across CoSt's, dequeued from work-queue

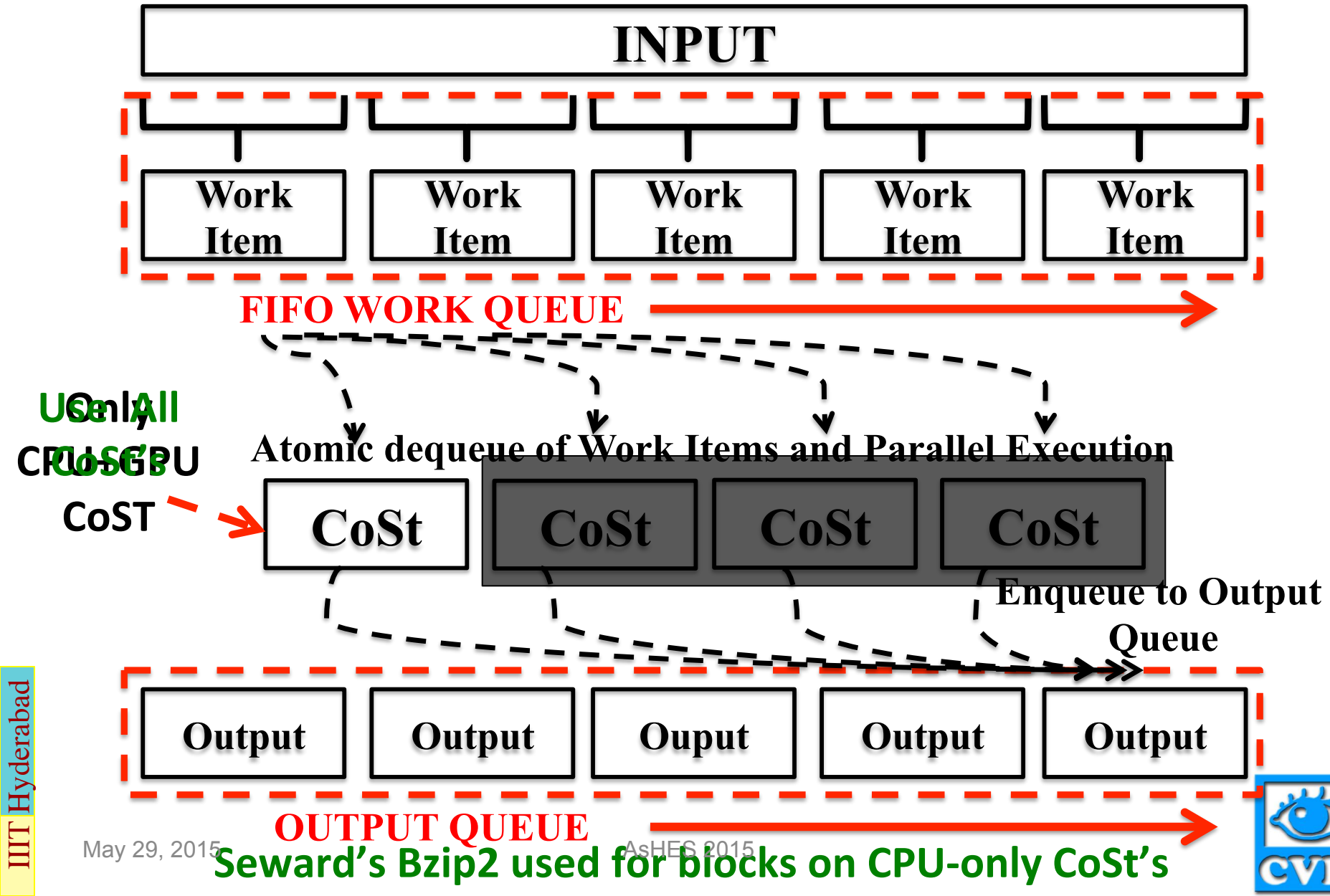


Hybrid BWC on CPU+GPU CoSt

- Patel et al. did all 3 steps on GPU, 2.78X slowdown
- Map appropriate operation to appropriate compute platform
- GPU for sorts of BWT, CPU does sequential merge, MTF, Huff
- Pipeline blocks such that CPU computation overlaps with GPU
- Throughput BWC = BWT, barring first and last block offset



Hybrid (CPU+GPU) BWC



Results: Hybrid BWC

Dataset (Size)	Block Size	(*i*) Compression time for our hybrid BWC (s), (*ii*) Compression time for CPU BWC (s) [Seward 2000], (*iii*) Compressed file size in MB's (same for both)			
		0% Perturb-ation	*0.01% Perturb-ation*	*0.1% Perturb-ation*	*1% Perturb-ation*
enwik8 (96MB)	900KB	**10.07**, 10.81, 27.66	**10.03**, 10.85, 27.70	**9.91**, 10.88, 28.09	**8.87**, 10.97, 31.32
	4.5MB	**7.29**, 13.12, **25.62**	**7.29**, 13.11, **25.67**	**7.31**, 13.10, **26.09**	**7.60**, 13.22, 29.36
	9MB	**8.31**, 15.23, **24.86**	**8.30**, 15.22, **24.91**	**8.33**, 15.82, **25.33**	**8.63**, 15.27, 28.66
wiki-xml (151MB)	900KB	**36.88**, 38.29, 15.29	**36.56**, 38.16, 15.39	**33.85**, 37.63, 16.19	**23.49**, 32.07, 21.89
	4.5MB	**30.42**, 60.78, **13.66**	**30.14**, 60.76, **13.77**	**26.97**, 60.55, **14.55**	**15.96**, 48.52, 19.82
	9MB	**31.51**, 80.76, **13.13**	**31.12**, 80.77, **13.24**	**27.62**, 79.94, **14.04**	**15.79**, 66.12, 19.07
linux-2.6.11.tar (199MB)	900KB	84.86, 24.93, 35.35	48.01, 24.69, 35.46	32.84, 23.21, 36.44	**22.78**, 22.17, 44.19
	4.5MB	133.54, 45.66, **33.10**	41.37, 44.02, **33.23**	**24.17**, 39.88, **34.26**	**14.24**, 26.65, 42.31
	9MB	196.64, 53.59, **32.51**	45.55, 51.77, **32.65**	**23.81**, 32.11, **33.69**	**14.37**, 29.64, 41.81
silesia.tar (203MB)	900KB	39.56, 29.65, 52.06	36.14, 29.69, 52.17	**28.98**, 29.32, 52.97	**23.06**, 27.46, 59.49
	4.5MB	34.60, 39.57, **50.06**	**29.52**, 39.63, **50.19**	**22.97**, 32.67, **51.03**	**16.81**, 36.07, 57.54
	9MB	36.10, 46.73, **49.57**	**28.85**, 46.92, **49.70**	**24.55**, 46.31, **50.55**	**17.74**, 41.94, 57.11

Compression Ratio improves with increase in Block Size

GPU runtime is better with larger blocks compared to CPU

GPU runtime improves with perturbation, CPU runtime stays the same

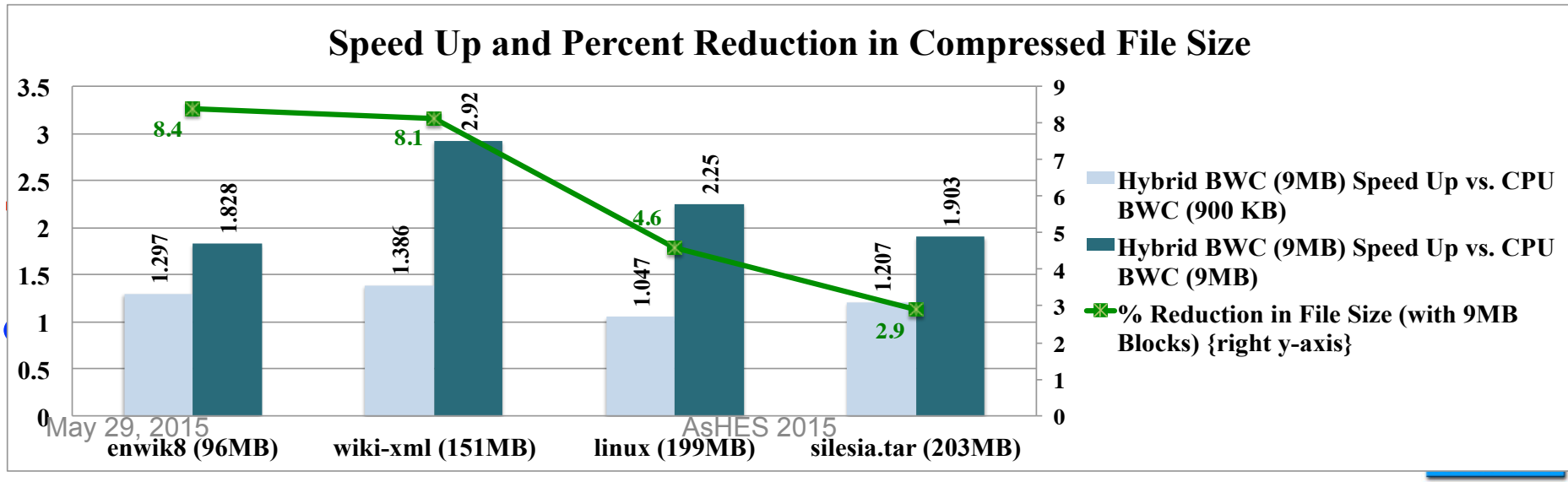
Compressed file size increases, but reasonable till 0.1% (< state-of-the-art)

Runtime & compressed file size better than state-of-the-art (Bzip2, 900KB)

Note, CPU does much less work using 900KB blocks, GPU uses 9MB.

Results: Hybrid BWC

Dataset (Size)	Block Size	(*i*) Compression time for our hybrid BWC (s), (*ii*) Compression time for CPU BWC (s) [Seward 2000], (*iii*) Compressed file size in MB's (same for both)			
		0% Perturb-ation	*0.01% Perturb-ation*	*0.1% Perturb-ation*	*1% Perturb-ation*
enwik8 (96MB)	900KB	**10.07**, 10.81, 27.66	**10.03**, 10.85, 27.70	**9.91**, 10.88, 28.09	**8.87**, 10.97, 31.32
	4.5MB	**7.29**, 13.12, **25.62**	**7.29**, 13.11, **25.67**	**7.31**, 13.10, **26.09**	**7.60**, 13.22, 29.36
	9MB	**8.31**, 15.23, **24.86**	**8.30**, 15.22, **24.91**	**8.33**, 15.82, **25.33**	**8.63**, 15.27, 28.66
wiki-xml (151MB)	900KB	**36.88**, 38.29, 15.29	**36.56**, 38.16, 15.39	**33.85**, 37.63, 16.19	**23.49**, 32.07, 21.89
	4.5MB	**30.42**, 60.78, **13.66**	**30.14**, 60.76, **13.77**	**26.97**, 60.55, **14.55**	**15.96**, 48.52, 19.82
	9MB	**31.51**, 80.76, **13.13**	**31.12**, 80.77, **13.24**	**27.62**, 79.94, **14.04**	**15.79**, 66.12, 19.07
linux-2.6.11.tar (199MB)	900KB	84.86, 24.93, 35.35	48.01, 24.69, 35.46	32.84, 23.21, 36.44	**22.78**, 22.17, 44.19
	4.5MB	133.54, 45.66, **33.10**	41.37, 44.02, **33.23**	**24.17**, 39.88, **34.26**	**14.24**, 26.65, 42.31
	9MB	196.64, 53.59, **32.51**	45.55, 51.77, **32.65**	**23.81**, 32.11, **33.69**	**14.37**, 29.64, 41.81
silesia.tar (203MB)	900KB	39.56, 29.65, 52.06	36.14, 29.69, 52.17	**28.98**, 29.32, 52.97	**23.06**, 27.46, 59.49
	4.5MB	34.60, 39.57, **50.06**	**29.52**, 39.63, **50.19**	**22.97**, 32.67, **51.03**	**16.81**, 36.07, 57.54
	9MB	36.10, 46.73, **49.57**	**28.85**, 46.92, **49.70**	**24.55**, 46.31, **50.55**	**17.74**, 41.94, 57.11



Results: All-Core BWC

NVIDIA GTX 580 (GPU) + Intel Core i7 920 (CPU)								
Dataset	**Total time for all-core BWC (CPU+GPU) (s), (#blocks processed by GPU / #total blocks)** Different number of CPU threads in addition to one CPU+GPU thread							Speedup (bold) vs. single CPU (underlined)
	0 CPU	**1 CPU**	**2 CPU**	**3 CPU**	**4 CPU**	**6 CPU**	**7 CPU**	
enwik8	8.4 (12/12)	6.5 (9/12)	5.8 (7/12)	4.7 (6/12)	4.7 (5/12)	4.6 (5/12)	**3.9 (4/12)**	4.05
wiki-xml	27.6 (18/18)	23.6 (13/18)	19.6 (10/18)	16.4 **(10/18)**	16.6 (8/18)	18.9 (7/18)	18.6 (6/18)	4.87
linux	23.8 (22/22)	14.3 (13/22)	10.88 (8/22)	9.3 (8/22)	9.1 (7/22)	**7.7 (5/22)**	7.9 (4/22)	4.16
silesia	24 (23/23)	16.8 (16/23)	13.3 (12/23)	12.6 (12/23)	12.7 (11/23)	11.4 (9/23)	**11.0 (8/23)**	4.20

Only Intel Core i7 920 (CPU)								
Dataset	**Total time for multi-core BWC (CPU only) (s)** Different number of CPU threads (No GPU involved)							Speedup (bold) vs. single CPU (underlined)
	1 CPU	**2 CPU**	**3 CPU**	**4 CPU**	**5 CPU**	**7 CPU**	**8 CPU**	
enwik8	15.8	9.1	6.4	5.0	5.0	4.9	**4.9**	3.22
wiki-xml	79.9	44.5	32.3	28.4	25.6	26.2	**26.1**	3.06
linux	32.1	17.6	13.3	10.7	10.0	10.3	**9.9**	3.24
silesia	46.3	30.9	21.5	21.4	21.1	**17.4**	18.4	2.66

- Using CPU CoSt's only: **3.06x** speedup
- Using all CoSt's (CPU and GPU): **4.87x** speedup

Conclusions

- Developed techniques for efficient use of all-core (CPU +GPU) systems
- String sort outperforms state-of-the-art significantly, adapts to future GPUs
- Our CPU+GPU hybrid GPU BWC shows a speed up for the first time on BWC using GPUs
- All-Core BWC shows improvement over using only the CPU or GPU cores for BWC
- Our results should encourage other developers to focus on development of fast end-to-end applications

Thank you!

Questions?

All codes are available for download at http://cvit.iiit.ac.in/ or https://web.engr.illinois.edu/~ardeshp2, CVIT/Personal Webpage

Please contact ardeshp2@illinois.edu or pjn@iiit.ac.in for more details

We thank the *'Indo-Israeli Project'* by *Department of Science and Technology* for partial financial support for this work

IIIT Hyderabad

CVIT